Weka-STPM: from trajectory samples to semantic trajectories

Luis Otavio Alvares[1], Andrey Luis Palma[2], Gabriel Oliveira[1], Vania Bogorny[3]

[1]Instituto de Informatica – Universidade Federal do Rio Grande do Sul (UFRGS)
Porto Alegre – Brazil

[2]Universidade Feevale – Novo Hamburgo – Brazil

[3]Instituto de Informatica e Estatistica – Universidade Federal de Santa Catarina (UFSC)
Florianopolis – Brazil

{alvares,gpoliveira}@inf.ufrgs.br,tietbohl@gmail.com,vania@inf.ufsc.br

Abstract. *Enormous quantities of trajectory data are collected from many sources, as GPS devices and mobile phones, as sequences of points. These data can be used in many application domains such as traffic management, urban planing, tourism, and bird migration. However, in most applications a higher level of abstraction should be used instead of sample points. In this paper we present an extension of the classical open source data mining toolkit Weka to support automatic trajectory data preprocessing in order to mining trajectories in a higher abstraction level. We propose Weka-STPM which is interoperable with all databases constructed under OGC specifications. We tested Weka-STPM with geographic databases and trajectory data stored into Postgresql/PostGIS, which is an open source GDBMS implemented according to OGC standards.*

1. Introduction

The use of mobile devices as GPS and cell phones have significantly increased in the last few years. This kind of devices and other technologies like RFID can generate sequences of space-time points, capturing the trajectories of moving objects. Trajectory data, acquired for operational level use, have increased enormously. However, very little has been done for the decisional level. One of the reasons is because trajectory data are usually generated as a sequence of (id, x, y, t) points, where id is the identification of the moving object, x and y are the geographic coordinates, and t is the moment when this point was collected. It is not easy to obtain useful information or knowledge from this kind of data, since the data themselves are very limited and have no direct link to other information, except the *owner* of the trajectory, through the moving object identification. Clearly, there is a need to consider trajectories from a higher level of abstraction, instead of simple (id, x, y, t) points.

Recently, Spaccapietra and others [Spaccapietra et al. 2008] proposed the first model to deal with trajectory data from a semantic point of view. This model is based on the concepts of *stops* and *moves*. Stops are the important places of a trajectory from an application point of view, where the moving object has stayed for a minimum amount of time. Moves are the subtrajectories between stops or between the starting point of the trajectory and the first stop or between the last stop and the ending point of the trajectory. Considering this model, a trajectory is a sequence of stops and moves. Figure 1 illustrates this concept. Figure 1(a) is a sample point trajectory with no semantics, while Figure 1(b)

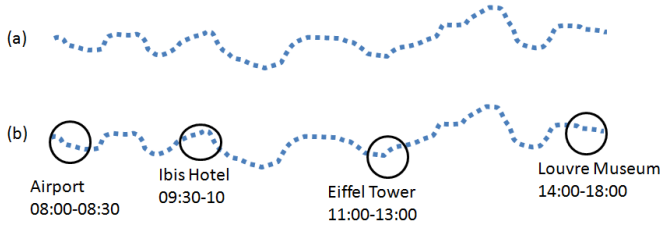

Figure 1. (a) Sample trajectory and (b) semantic trajectory

represents the same trajectory considering a touristic application, where we can see that the trajectory starts in an airport and stays there from 8AM to 8:30AM. Then, it goes to Ibis Hotel where it arrives at 9:30AM and it leaves at 10AM. After that, the moving object goes to the Eiffel Tower, where it stays from 11AM to 1PM. Finally, it goes to the Louvre Museum and stays there from 2PM to 6PM.

To transform a sample trajectory into a semantic one (sequence of stops and moves) is not an easy task. To do it, one should consider several aspects as the type of application and what geographic background information in the region of the trajectory is important for the specific problem in hand.

In order to help in the solution of this problem, this paper presents a free software called Weka-STPM (Semantic Trajectories Preprocessing Module) that has been constructed into Weka [Frank et al. 2005] to preprocess raw trajectories in order to transform them into semantic trajectories. As a module of Weka, it allows the user to directly apply the several mining algorithms available in Weka to mine semantic trajectories. Weka-STPM is the first tool for semantic trajectory preprocessing for data mining.

The remainder of the paper is structured as follows. Section 2 presents two methods to generates semantic trajectories. Section 3 describes the Weka-STPM module and Section 4 presents the conclusion of the paper and future works.

2. Trajectory data preprocessing in STPM

To prepare trajectory data to data mining, the main step is to add semantics to these trajectories according to an application point of view. We do that by using the concepts of stops and moves. Two algorithms have been developed so far. The first one, introduced in [Alvares et al. 2007], considers the intersection of a trajectory with the user-specified relevant feature types (relevant object type or candidate stop) for a minimal time duration, which is called IB-SMoT (Intersection-Based Stops and Moves of Trajectories).

In general words, the algorithm verifies for each point of a trajectory T if it intersects the geometry of a relevant feature type R_C. In affirmative case, the algorithm looks if the duration of the intersection is at least equal to a given threshold Δ_C. If this is the case, the intersected candidate stop is considered as a stop, and this stop is recorded. If a point does not belong to a subtrajectory that intersects a candidate stop for Δ_C, it will be part of a move.

Figure 2 (a) illustrates this method. In this example, there are four candidate stops with geometries R_{C_1}, R_{C_2}, R_{C_3}, and R_{C_4}. Let us consider a trajectory T represented by the space-time points sequence $\langle p_0, \ldots, p_{15} \rangle$, and $t_0, \ldots, t_{15}$ are the time points of T.

First, T is outside any candidate stop, so it starts with a move. Then T enters R_{C_1} at point p_3. Since the duration of staying inside R_{C_1} is long enough, R_{C_1} is the first stop of T, and $\langle p_0, \ldots, p_3 \rangle$ is its first move. Next, T enters R_{C_2}, but for a time interval shorter than Δ_{C_2}, so this is not a stop. We therefore have a move until T enters R_{C_3}, which fulfills the requests to be a stop, and so R_{C_3} is the second stop of $T$and $\langle p_5, \ldots, p_{13} \rangle$ is its second move.

The second algorithm is called CB-SMoT (Clustering-Based Stops and Moves of Trajectories) [Palma et al. 2008], and is a clustering method based on the variation of the speed of the trajectory. The intuition of this method is that the parts of a trajectory in which the speed is lower than in other parts of the same trajectory, correspond to interesting places. CB-SMoT is a two-step algorithm. In the first step, the slower parts of one single trajectory are identified, using a spatio-temporal clustering method. In the second step, the algorithm identifies where these potential stops (clusters) found in the previous step are located, considering the candidate stops. In case that a potential stop does not intersect any of the given geographic objects, it still can be an interesting place. In order to provide this information to the user, the algorithm labels such places as *unknown stops*. Unknown stops are interesting because although they may not intersect any relevant spatial feature type given by the user, a pattern can be generated for unknown stops if several trajectories stay for a minimal amount of time at the same unknown stop. In this case, the user may investigate what this unknown stop is.

Figure 2 (b) illustrates the method CB-SMoT. Considering the trajectory $T = \langle p_0,\ p_1,\ \ldots,\ p_n \rangle$ represented in Figure 2 (b), the first step is to compute the clusters. Suppose that T has 4 potential stops, the clusters G_1, G_2, G_3 and G_4, represented by ellipses. In this example the user has specified 4 candidate stops, identified by the rectangles R_{C1}, R_{C2}, R_{C3} and R_{C4}. The cluster G_1 intersects the candidate stop R_{C1} for a time greater than Δ_{c1}, then the first stop of the trajectory is R_{C1}. The same occurs with the cluster G_2, considering R_{C3}, which is the second stop of the trajectory. The clusters, G_3 and G_4 do not intersect any candidate stop. Therefore, G_3 and G_4 are unknown stops.

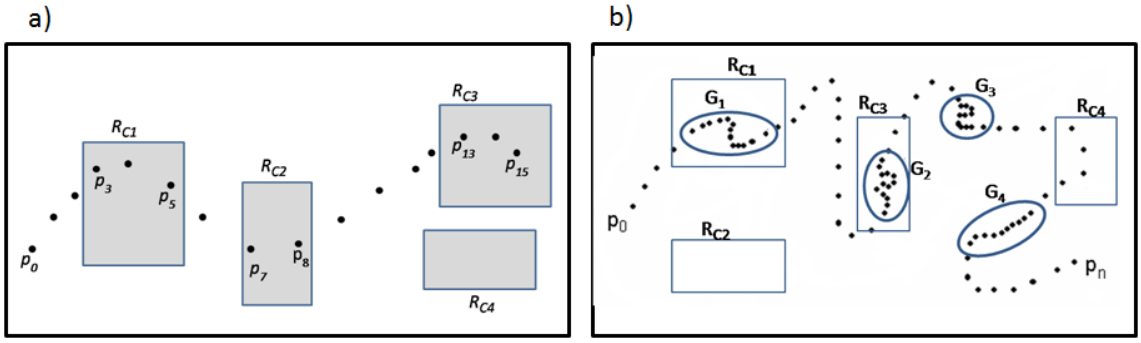

Figure 2. a) Example of the IB-SMoT method, and b) Example of the CB-SMoT method

The two methods cover a relevant set of applications. IB-SMoT is interesting in applications where the speed is not important, like tourism and urban planning. In this kind of application, the presence or the absence of the moving object in relevant places is more important. However, in other applications like traffic management, CB-SMoT, which is based on speed, would be more appropriate.

3. Weka-STPM

Weka is a free and open source non-spatial data mining toolkit developed in Java. It has a non-spatial data preprocessing module named weka.Explorer, where data can be obtained from a database, a web site, or an arff file(input text file in the format required by Weka). The module STPM is fully integrated into Weka in order to automatically access the database and add semantics to trajectory data. Weka-STPM is an extension of Weka for spatio-temporal data. It is interoperable with all databases constructed under Open GIS Consortium (OGC) specifications [OGC 2008].

In order to support STPM we extended the Weka database connection interface. We added the button Trajectories, shown in Figure 3 (a), which calls the STPM module. As can be observed in Figure 3 (c), the user provides the database schema and STPM loads all geographic database tables to the boxes *Trajectory Table* and *Relevant Features*. This allows the user to choose the target trajectory table and the spatial feature types of interest (candidate stops) with the respective minimum time duration, in order to be considered a stop (*RF Min Time*). The parameter *User Buffer* is the radius (size) of the zone around relevant features represented by points or lines, to overcome spatial uncertainty. Besides, the user can choose the *Method* to be used to generate the semantic trajectories (IB-SMoT or CB-SMoT) and define the respective parameters.

The software has some limitations concerning the trajectory table. This table should have an attribute called *Tid*, of type integer, that is used as the trajectory identification; and the attribute that stores the time information should be called *time*, and be of type timestamp. In order to either generate or transform these attributes we added the button *Trajectory Table Config* that opens the window shown in Figure 3 (b). It presents some examples of scripts that can be adapted to do these transformations and can be executed from this interface. Furthermore, there is a *Clean Trajectories* button (Figure 3 (c)) that removes trajectories with less than a specified amount of points, in order to remove noise.

The output of the STPM module are relations of stops and moves stored in the database as well as two files: *stops.arff* and *moves.arff*. The schema of the stop relation has the following attributes:

```
STOP (Tid integer, Sid integer, SFTname varchar, SFid integer,
      startT timestamp, endT timestamp)
```

where:

- *Tid*: is the trajectory identifier.
- Sid: is the stop identifier. It is an integer value starting from 1, in the same order as the stops occur in the trajectory. This attribute represents the sequence as stops occur in the trajectory.
- *SFTname*: is the name of the relevant spatial feature type (geographic database relation) where the moving object has stayed for the minimal amount of time.
- *SFid*: is the identification of the instance (e.g. Ibis) of the spatial feature type (e.g. Hotel) in which the moving object has stopped.
- *startT*: is the time in which the stop has started, i.e., the time that the object enters in a stop.
- *endT*: is the time in which the moving object leaves the stop.

The relation of moves has the following schema, with four attributes more than the stop relation:

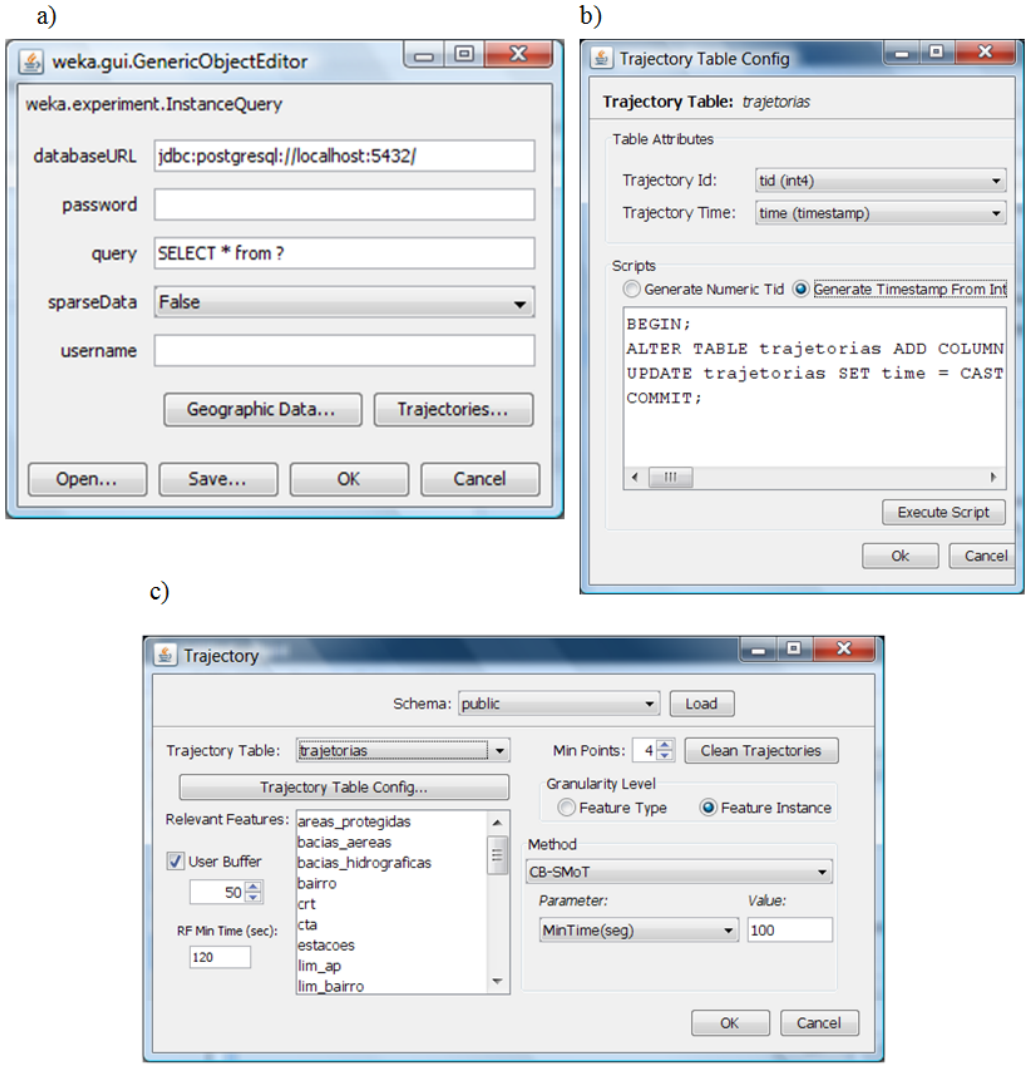

Figure 3. STPM interface

```
MOVE (Tid integer, Mid integer, SFT1name varchar, SF1id integer,
      SFT2name varchar, SF2id integer, startT timestamp,
      endT timestamp, the_move multiline )
```

where:

- *Mid*: is the identifier of the move in the trajectory. It starts with 1, in the same order as the moves occur in the trajectory.
- *SFT1name and SFT2name* : are the names of the spatial feature type in which the move respectively starts and finishes.
- *SF1id and SF2id*: are the identifier (feature instance) of the start and end stop of the move.
- *the_move*: is the set of points that corresponds to the spatial properties of a move.

So far, we have tested the prototype with data stored in a Postgresql/PostGIS database. We have performed experiments with real trajectory data collected in the city of Rio de Janeiro.

4. Conclusions and Future Work

The data generated by mobile devices are raw data that, by their nature, are very difficult to be used for decision making. In order to explore these data for different application domains, two methods have been developed to enrich these data with domain-specific geographic information. This paper presented Weka-STPM, an open source tool that implements these two methods to deal with trajectories of moving objects from a higher abstraction level. It is the first tool to integrate trajectories and geographic data, both stored in a Postgres/PostGIS database.

With Weka-STPM the user is able to automatically add semantic geographic information to trajectories, in a preprocessing step, and then exploit several data mining methods available in Weka for data mining.

This work was possible thanks to the community that develops open source software. Weka is implemented in a way that allows its extension with new modules valuable and useful for several communities and application domains.

As future ongoing work we are extending Weka-STPM with new preprocessing methods and a graphical GUI to visualize in a map the spatial entities and the generated stops and moves.

Weka-STPM is available for download at *www.inf.ufrgs.br/∼alvares/software* and *www.inf.ufsc.br/∼vania/software*.

5. Acknowledgements

This work was partially supported by the Brazilian agencies CNPq (307588/2008-4) and FAPESC (CP005/2009).

References

Alvares, L. O., Bogorny, V., Kuijpers, B., de Macedo, J. A. F., Moelans, B., and Vaisman, A. (2007). A model for enriching trajectories with semantic geographical information. In *ACM-GIS*, pages 162–169, New York, NY, USA. ACM Press.

Frank, E., Hall, M. A., Holmes, G., Kirkby, R., Pfahringer, B., Witten, I. H., and Trigg, L. (2005). Weka - a machine learning workbench for data mining. In Maimon, O. and Rokach, L., editors, *The Data Mining and Knowledge Discovery Handbook*, pages 1305–1314. Springer.

OGC (2008). Opengis standards and specifications. Available at: http://http://www.opengeospatial.org/standards. Accessed in August 2008.

Palma, A. T., Bogorny, V., Kuijpers, B., and Alvares, L. O. (2008). A clustering-based approach for discovering interesting places in trajectories. In *ACMSAC*, pages 863–868, New York, NY, USA. ACM Press.

Spaccapietra, S., Parent, C., Damiani, M. L., de Macedo, J. A., Porto, F., and Vangenot, C. (2008). A conceptual view on trajectories. *Data and Knowledge Engineering*, 65(1):126–146.